

**IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division**

DAEDALUS BLUE, LLC,

Plaintiff,

v.

MICROSTRATEGY INCORPORATED,

Defendant.

Case No. 20-cv-01326

JURY TRIAL DEMANDED

DAEDALUS BLUE, LLC’S COMPLAINT FOR PATENT INFRINGEMENT

NOW COMES the Plaintiff, Daedalus Blue, LLC “(Daedalus)”, pursuant to Fed. R. Civ. P. 4 and 28 U.S.C. § 1338(a), and for its Complaint against Defendant MicroStrategy Incorporated (“MicroStrategy”) hereby alleges as follows:

INTRODUCTION

1. The novel inventions disclosed in the Asserted Patents in this matter were invented by International Business Machines Corporation (“IBM”). IBM is and has been a pioneer in the computing world. Every year, IBM spends billions of dollars on research and development to invent, market, and sell new technology, and IBM obtains patents on many of the novel inventions that come out of that work, including the Asserted Patents. The two patents asserted in this case are the result of the work from six different IBM researchers.

2. Over the years, the inventions claimed in the Asserted Patents have been licensed to many companies, including Amazon Web Services and Oracle Corporation.

THE PARTIES

3. Daedalus is the current owner and assignee of the Asserted Patents.

4. Daedalus is a Delaware limited liability company with its principal place of business located at 51 Pondfield Road, Suite 3, Bronxville, NY 10708.

5. Defendant MicroStrategy is a Delaware corporation with a principal place of business at 1805 Towers Crescent Plaza, Tysons Corner, Virginia 22182.

6. MicroStrategy conducts business in Virginia and in the Eastern District of Virginia, as set forth below.

JURISDICTION AND VENUE

7. This is an action arising under the patent laws of the United States, 35 U.S.C. § 101, *et seq.* Accordingly, this Court has subject matter jurisdiction pursuant to 28 U.S.C. §§ 1331 and 1338(a).

8. This Court has personal jurisdiction over Defendant MicroStrategy because, *inter alia*, Defendant (1) has substantial, continuous, and systematic contacts with this State and this judicial district; (2) owns, manages, and operates facilities in this State and this judicial district; (3) enjoys substantial income from its operations and sales in this State and this judicial district; (4) employs Virginia residents in this State and this judicial district; and (5) solicits business and market products, systems and/or services in this State and judicial district including, without limitation, those related to the infringing accused products.

9. Venue is proper in this District pursuant to at least 28 U.S.C. § 1319(b)-(c) and § 1400(b), at least because Defendant MicroStrategy, either directly or through its agents, has committed acts within this judicial district giving rise to this action, and continue to conduct

business in this district, and/or has committed acts of patent infringement within this District giving rise to this action.

FACTUAL ALLEGATIONS

The Asserted Patents

10. The IBM inventions contained in the Asserted Patents in this case relate to groundbreaking improvements to computer functionality and computer security. The techniques IBM developed are described in the Asserted Patents and relate to computer networks and have particular application in the fields of Enterprise security, access control, and data processing and analytics as will be further described below.

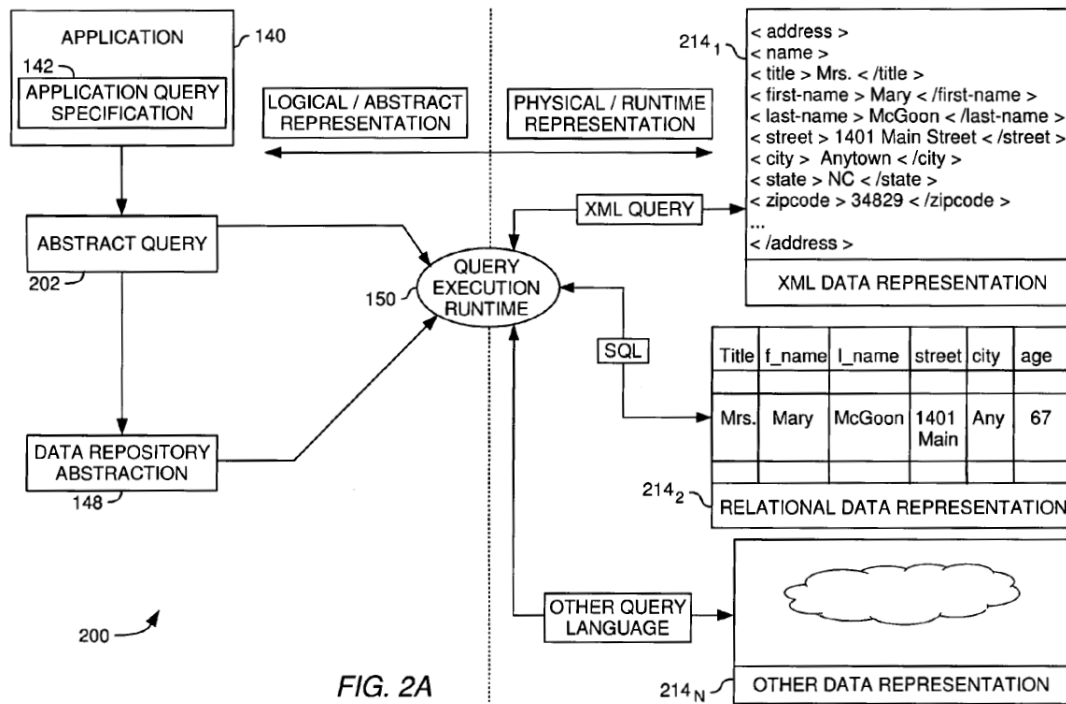
U.S. Patent No. 8,341,172

11. On December 25, 2012, the U.S. Patent and Trademark Office duly and lawfully issued United States Patent No. 8,341,172 (“the ’172 Patent”), entitled “Method and system for providing aggregate data access.” A true and correct copy of the ’172 Patent is attached hereto as **Exhibit 1**.

12. Daedalus is the owner and assignee of all right, title, and interest in and to the ’172 Patent, including the right to assert all causes of action arising under said patent and the right to any remedies for infringement of it.

13. The ’172 Patent describes, among other things, novel systems and methods that improve the functioning of a computer, including improvements to the way in which information storage and retrieval systems store and access data through improved means of generating aggregate data values from across one or more data sources, and merging aggregate and non-aggregate data values. The inventive technological improvements described in the ’172 Patent solved then-existing problems in the field of computer database operations. For example, the

'172 Patent describes one issue that can arise with data queries when a change occurs in an underlying database structure or schema. Databases may be structured in a variety of ways and the database "schema" generically refers to a particular arrangement of data. For example, XML is one type of schema; another prevalent schema is the relational database schema. The relational database is a tabular database in which data is defined in a manner in which it can be reorganized and accessed a variety of ways. In order to retrieve or update data in a database, an entity (such as an application, the operating system, an administrator, or a user) issues a query. A query is a set of commands structured in a particular programming language, specifying columns, tables and the various relationships between them relevant to the query. (Ex. 1, at 1:20-60). The standard query language used for manipulating data in a relational database is called SQL (Structured Query Language). Fig. 2A of the '172 Patent, for example, shows that at the physical/runtime representation, a relational data schema (214₂) is queried using SQL, an XML schema (214₁) is queried using XML, etc. As also depicted in Fig. 2A, computer systems use a logical/abstract representation to communicate and translate between an application (with code at the application layer) and the physical representation of data (physical data layer):



14. Prior to the invention of the '172 Patent, a drawback in database query applications was their close relationship with a given database schema. For example, an application relying on SQL would be dependent on a particular relational schema because specific database tables, columns and relationships would be references within a SQL query representation. Thus, the application-schema relationship would make it difficult to support the application if changes were made to the underlying schema. Additionally, migrating the application to an alternative underlying data representation was inhibited. (Ex. 1, at 1:53-67). For example, changes in the underlying relational data model required changes to the SQL foundation, requiring an application designer to either avoid application maintenance that changed the underlying data model or to change the application to reflect the changes in the underlying relational model, perhaps even requiring separate versions of the application to account for requirements driven by each relational schema. (*Id.* at 2:1-10). Prior to the

invention of the '172 Patent, for an application to work with alternative data representations, an engineer might need to rewrite the application's data management layer to use additional data access methods. Generally, in conventional systems, software encapsulation was used to address the foregoing problems, which involves using a software interface or component to encapsulate access methods to a particular underlying data representation. (*Id.* at 2:16-20). However, this approach was costly, inflexible, and highly specialized and required software to be written (including a code update, application build and deployment cycle) when a new set of data is accessed, or a new pattern of data was desired. (*Id.* at 2:17-38). Eliminating the need to encapsulate access methods and removing the administrator's role allowed the '172 Patent invention to provide more flexibility during the overall technological data handling processes.

15. Prior to the invention of the '172 Patent, a computer database system might incorporate a data abstraction layer that sits between an application and the underlying physical data. The data abstraction layer defines a collection of logical fields that are loosely coupled to the underlying physical mechanisms storing the data. (Ex. 1 at 2:42-44). One difficulty encountered in constructing an abstraction layer, however, is representing data derived from multiple rows of data stored in an SQL table (e.g., columnar data). (*Id.* at 2:1-3). Prior to the '172 Patent invention, in this example, a database administrator familiar with the underlying physical database schema and language (e.g., SQL) might need to create individual SQL views that perform aggregate calculations then specify these as a data source that the abstract model may query and join with other results. That is, in addition to the database administrators' duties, she also had to create a static view for each different aggregation. Also, users were not able to dynamically alter the rows or columns involved in the aggregation (e.g., changing the query from determining the average age of the first 10 females to determining the average age of the first 5

males), rather users solely relied on the database administrator to perform each aggregation from a statically created view. This causes a problem particularly in a distributed environment where underlying data sources may change frequently, and statically created views may also reference underlying data that is no longer available to respond to a query request. (*Id.* at 2:54-3:5).

16. The '172 Patent overcomes these drawbacks and improves the functioning of computer database systems and services, for example, by disclosing novel and inventive systems which allow a computer to return aggregate values for related groupings of rows joined to non-aggregate data without requiring the maintenance of a static view for each aggregation.

Composing queries from aggregate expressions and set(s) of input data by using database abstraction techniques now allows users, without requiring an administrator's intervention each time an aggregation is run or a schema is altered, to dynamically apply conditions on the results and place restrictions on the input for each aggregation. Improving the computer's capabilities by removing reliance on another individual seeing the request for, then executing each desired aggregation results in faster and more accurate data processing, and also solves the problem in a distributed environment where static tables may reference outdated or unavailable data. Such techniques further do not require that a user be familiar with the underlying physical data representation, thus improving the accessibility of the system.

17. In one embodiment, the '172 Patent describes an Abstract Derived Entity (ADE) which is a data object present in the data abstraction layer that is referenced by an access method as though it were a table. In one example, when a query specifies a selection or a result for a field defined over an ADE, the ADE is converted to a derived table at the time the abstract query is converted to an SQL query and the derived table may then be joined with other tables referenced in the SQL query. (Ex. 1 at 5:25-38). By combining logical fields that use aggregate

access methods with an ADE, users may create complex queries that examine or discover trends in existing data. The abstract derived entity is used to dynamically generate a derived table during query execution that does not exist in the underlying physical data sources. Doing so allows users to construct abstract queries that alter the data used in the aggregation calculation based on, for example, the filtering expression of a filtered field and may reveal correlations that are not readily apparent from the test data alone. (*see id.* at 13:21-50).

18. The '172 Patent gives an example of how the invention allows for discovery of trends in existing data that were not previously available. For example, Figure 2C depicts a query that allows for identification of individuals whose glucose levels showed a rising trend (216) when taking the drug named MK-767 (218).:

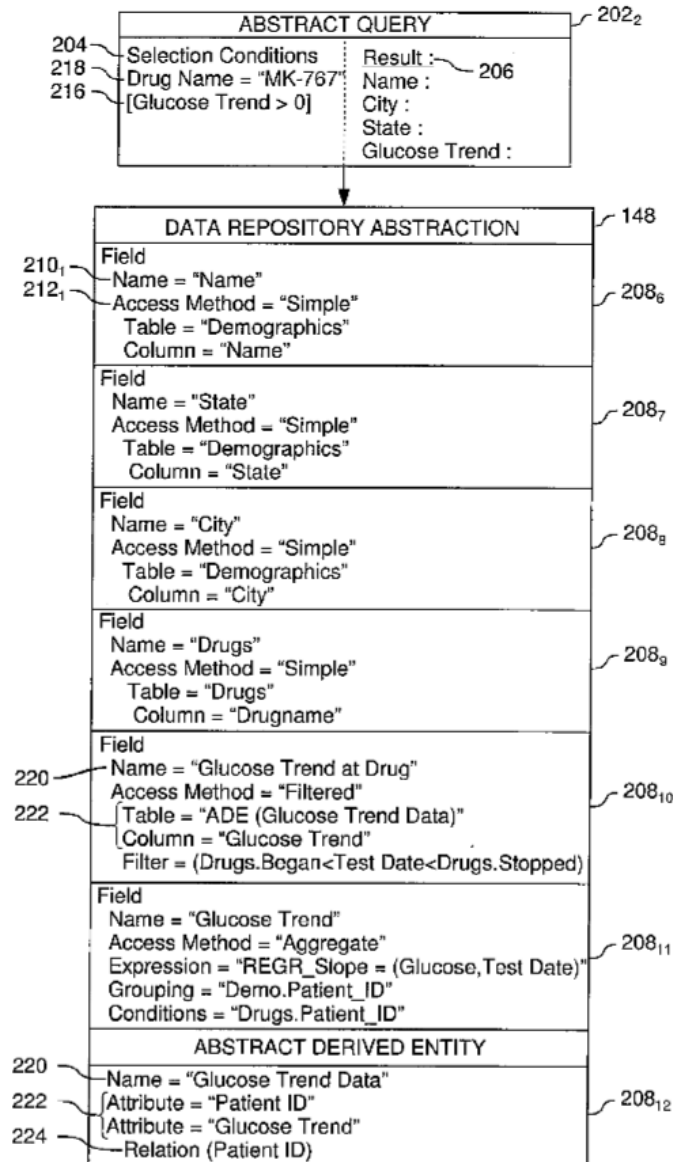


FIG. 2C

Field specification 208₁₁ is an example of a logical field that maps to data using an aggregate access method. The logical field "glucose trend" maps to an aggregate value calculated from the "glucose" and "testdate" columns of an underlying Lab Tests table shown in Fig. 3. Aggregate access methods return an aggregate value calculated from the row values of a relational table. An aggregate access method includes an expression used to calculate the aggregate value returned by the logical field. Illustratively, field specification 208₁₁ includes the expression

“REGR_SLOPE (glucose, testdate).” The parameters included with the expression are the named columns of a physical table (or an ADE derived table) used to calculate the aggregate value. In this example, the expression uses parameters taken from the Lab Tests table. In addition to the expression, an aggregate access method may provide grouping conditions used to join aggregate values to non aggregate data. As illustrated, field specification 208₁₁ includes the grouping condition of “patient ID.” Thus, an aggregate value is calculated from the rows of the Lab Tests tab for each patient ID. (Ex. 1 at 12:43-61). The aggregate access method illustrated in 208₁₁ determines a glucose trend from the slope of the line generated from a regression function applied to a set of data points. Each data point provided to the regression function is composed as (Test Date, Glucose Level). A user may then include this logical field in an event profile such as “(glucose trend>0).” Applying this event profile to other conditions in an abstract query may be used to uncover trends from the underlying data. A positive value for a glucose trend indicates that glucose levels are rising. When this event profile is applied to a condition such as “Drugs Taken=MK-767” the effect is to identify patients whose glucose levels exhibited a rising trend when the patient was taking drug MK-767. (*Id.* at 13:9-21).

19. The novel features of the invention are recited in the claims. For example, Claim 1 of the '172 Patent recites:

A system for generating aggregate data values from data stored in a data source, comprising:

a processor;

a database service available in a network environment, the database service comprising (a) a data source, (b) an abstract data layer, wherein the abstract data layer comprises a plurality of logical fields used to compose an abstract query to query the data source, and for each logical field, providing an access method specifying at least a method for accessing the data, wherein at least one logical field specifies an aggregate access method, wherein the aggregate access method specifies a set of input data and an

expression for determining an aggregate data value from the set of input data; and

(c) a runtime component configured to process an abstract query that includes the at least one logical field by (i) retrieving a definition for the aggregate access method, (ii) determining aggregate data values according to the definition, (iii) merging the aggregate data values with query results obtained for logical fields, other than the at least one logical field, included in the abstract query, and (iv) returning the results to the requesting entity.

(Ex. 1 at 22:7-28). Claim 1 of the '172 Patent describes claim elements, individually or as an ordered combination, that were non routine and unconventional at the time of the invention in 2004 and an improvement over prior art, as it provided a way (not previously available) to perform dynamic data aggregation without the need for an administrator, and a way to create, manage and use a database abstraction model that provides an aggregated abstraction for an underlying physical database. For example, when accessing, generating, or viewing aggregate values, it was unconventional to allow users to restrict the rows included in a particular aggregation or to apply conditions to the aggregation results (e.g., average blood pressure of select patients or for specific dates is requested from "Lab Test" data). (*Id.* at 20:1-49). Further, it was unconventional to provide an aggregate access method that specifies a set of input data and an expression for determining an aggregate data value from the set of input values.

U.S. Patent No. 9,032,076

20. On May 12, 2015, the U.S. Patent and Trademark Office duly and lawfully issued United States Patent No. 9,032,076 ("the '076 Patent"), entitled "Role-Based Access Control System, Method and Computer Program Product." A true and correct copy of the '076 Patent is attached hereto as **Exhibit 2**.

21. Daedalus is the owner and assignee of all right, title, and interest in and to the '076 Patent, including the right to assert all causes of action arising under said patent and the right to any remedies for infringement of it.

22. The '076 Patent describes, among other things, novel techniques that improve the methods for restricting and granting user access to resources, which results in enhanced system security. These inventive technological improvements solved then-existing problems in the field of computer networks and security authorizations that control access to various resources. For example, the '076 Patent explains that, in a computer network, a company may want to control which users have access to particular resources such as servers or storage spaces. Thus, the system usually includes a resource management function which synchronizes and manages access to such resources. The resource manager within the server systems is often assigned the task of security and access control such that users requesting secure data from the resources may be allowed or denied access to that data. Traditionally, access would be provided through the use of access control lists (ACL), whereby users are associated with specific permissions to access or to interact with various resources. In a classical role-based access control model, the ACLs are used to provide users with specific permissions to access or to interact with various resources. In such systems, whenever a permission within an ACL changes, the ACL must be recreated with the changed permission, thereby lacking the possibility to enforce different access control constraints on individual resource instances. To overcome this problem, solutions prior to the invention of the '076 Patent used extensions to the classical model defining roles to be sets of permissions on individual resources (resource-level role-based access control (RRBAC)). However, then-existing access control models did not provide instance level resource protection. (See Ex 2, at 1:50-2:6). Furthermore, then-existing RRBAC models restrict a role domain

associated to a specific role instances to protect exactly one sub-hierarchy of resources of the protected resource hierarchy. (*Id.* at 4:44-48).

23. The '076 Patent overcomes these drawbacks and improves the functioning of a computer system, for example, by disclosing novel and inventive systems in which access to system resources is controlled by assigning roles and super roles to groups of users. The super roles are defined by grouping a set of role instances (permissions on individual resources), wherein each super role contains all permissions assigned to each of the role instances in the grouped set of role instance. The super roles are modified by adding or removing role instances in the grouped set of role instances. In one aspect, the patent discloses an improved role-based access control system “which comprises a role definition system for defining roles to be sets of permissions on individual resources thus forming role instances, respectively, and a super role definition system for defining at least one super role by grouping a set of role instances, wherein each super role contains all permissions contained in the grouped role instances.” (Ex. 2 at 2:30-35).

24. For example, Figure 2 of the '076 depicts a schematic diagram of an embodiment of a role-based access control system according to the '076 invention:

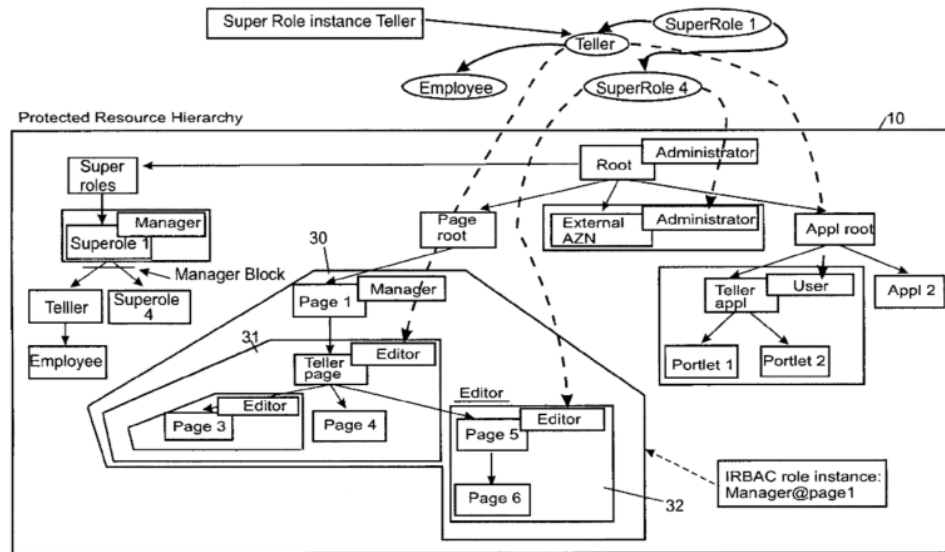


FIG. 2

In general, the small rectangular nodes of the hierarchical tree 10, each represent a resource.

Above the hierarchical tree 10, four super roles are shown, a super role, called “SuperRole 1,” a super role, called “Teller,” a super role, called “SuperRole 4,” and a super role, called “Employee”. The super roles are arranged in a hierarchical manner. Super role “SuperRole 1” contains super role “Teller” and super role “SuperRole 4,” while super role “Teller” contains super role “Employee.” That means that all permissions coupled with super role “Teller” and “SuperRole 4” are also assigned to super role “SuperRole 1.” All permissions assigned to super role “Employee” are also assigned to super role “Teller” and thus to super role “SuperRole 1.” (Ex. 2 at 6:48-60.)

25. The claims of the '076 Patent are directed to specific techniques that nest super roles wherein each super role contains all permissions assigned to each of the role instances in the grouped set of role instances. The '076 Patent further defines at least one super role wherein each super role is nested according to a plurality of properties including a name, a parent role, the set of role instances, and an externalization state. Through the patented invention, access

control administration complexity is reduced, as well as costs and errors that could result in unintended access control configurations. Moreover, access control delegation flexibility is improved and the disclosed mechanisms can reduce the set of authorized people necessary at a specific point in time thereby improving overall security and auditability. The '076 Patent also provided a way (not previously available) to modify super roles by adding and/or removing role instances from the grouped set of role instances.

26. The novel features of the invention are recited in the claims. For example, Claim 6 of the '076 Patent recites:

A role-based access control method, comprising:

defining roles to be sets of permissions on individual resources, thus forming role instances, respectively;

assigning at least one set of role instances to at least one group and assigning the at least one group to at least one super role; and

nesting each super role according to a plurality of properties including a name, a parent role, the set of role instances, and an externalisation state,

wherein each super role is modified by adding or removing the role instances from the at least one group.

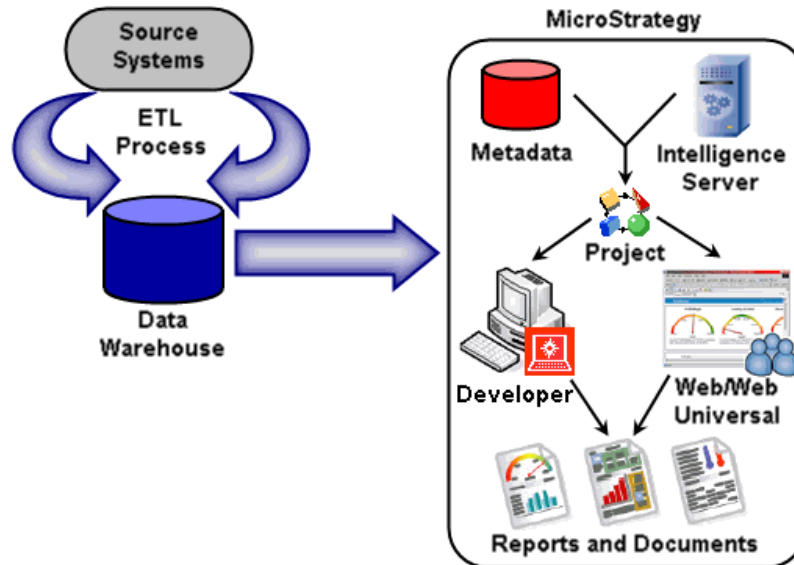
(Ex. 2 at 14:20-30). Claim 6 of the '076 Patent describes claim elements, individually or as an ordered combination, that were non routine and unconventional at the time of the invention in 2005 and an improvement over prior art, as it provided a way (not previously available) to operate a role based access control system. For example, in a system for defining roles, it was unconventional to define at least one super role wherein each super role is nested according to a plurality of properties including a name, a parent role, the set of role instances, and an externalization state.

27. The '172 and '076 Patents are referred to hereinafter as “the Asserted Patents.”

28. Each of the Asserted Patents are presumed valid under 35 U.S.C. § 282.

MicroStrategy's Use of the Patented Technology

29. MicroStrategy is a company that sells enterprise analytics software and services. MicroStrategy's core product offering is the MicroStrategy Platform. The MicroStrategy Platform incorporates various analytics and advanced reporting tools, including, for example, the Query Builder functionality (which is incorporated in the MicroStrategy Desktop and/or MicroStrategy Developer components/products), as well as the Intelligence Server functionality. The following diagram shows how Platform components function together:



(See BI Architecture and the MicroStrategy Platform, available at https://www2.microstrategy.com/producthelp/current/ProjectDesignGuide/WebHelp/Lang_1033/Content/ProjectDesign/The_MicroStrategy_platform.htm).

30. The MicroStrategy Cloud Environment (MCE) is a cloud subscription service hosted and managed by MicroStrategy. The MicroStrategy Platform can be deployed on-premises or in the cloud, including in the MCE. MicroStrategy continues to see growth in cloud subscriptions and intends to migrate more customers to the cloud over the next few years. (See

2Q 2020 Financial Results, available at <https://ir.microstrategy.com/static-files/bd1abfa7-786d-4934-b988-7da3b81daff6>).

31. MicroStrategy identifies IBM as one of its key competitors in the analytics market. (*See e.g.*, MicroStrategy Incorporated 2019 ANNUAL REPORT, available at <https://ir.microstrategy.com/static-files/21dbace7-79c6-47c0-bce5-ea40ce2368a7>).

32. On information and belief, MicroStrategy makes, uses, sells, and/or offers to sell in the United States, and/or imports into the United States various methods and/or products relating to Enterprise security, access control, and analytics and data processing which infringe the Asserted Patents.

33. MicroStrategy makes, uses, sells, and/or offers to sell in the United States, and/or imports into the United States the MicroStrategy Platform.

34. On information and belief, the MicroStrategy Platform has also been referred to as the MicroStrategy Analytics Platform.

35. The MicroStrategy Platform includes MicroStrategy advanced reporting tools which provide analytics services, such as the exemplary Query Builder, and all reasonably similar products (“the Advanced Reporting Tools”). (*See* Advanced Reporting Guide, <https://www2.microstrategy.com/producthelp/Current/manuals/en/AdvancedReporting.pdf>).

36. MicroStrategy’s Advanced Reporting Tools deliver a comprehensive set of analytics capabilities, including generating aggregate data values from data stored in a database. (*See* Introduction to advanced Reporting, available at https://www2.microstrategy.com/producthelp/Current/AdvancedReportingGuide/WebHelp/Lang_1033/Content/Intro_to_Advanced_Report.htm).

37. MicroStrategy's Advanced Reporting Tools include the Query Builder Editor which allows users to "define the SQL queries to run against imported database tables." (*See* Query Builder Editor, available at https://www2.microstrategy.com/producthelp/Current/ReportDesigner/WebHelp/Lang_1033/Content/query_builder_editor.htm). The Query Builder Editor is also available in other products including the MicroStrategy Platform version 10.4, MicroStrategy Platform 2019, and MicroStrategy Platform 2020. (*See, e.g.,* Advanced Reporting Guide version 10.4, available at <https://www2.microstrategy.com/producthelp/10.4/manuals/en/AdvancedReporting.pdf>; Accessing the Query Builder Editor, available at https://www2.microstrategy.com/producthelp/2019/ReportDesigner/WebHelp/Lang_1033/Content/ReportDesigner/steps_to_access_the_query_builder_editor.htm; Query Builder Editor, available at https://www2.microstrategy.com/producthelp/Current/ReportDesigner/WebHelp/Lang_1033/Content/query_builder_editor.htm).

38. Using a Query Builder Editor, a graphical user interface, users can compose abstract queries without having to write any SQL. (*See* MicroStrategy Advanced Reporting Guide, at 773, available at <https://www2.microstrategy.com/producthelp/current/manuals/en/AdvancedReporting.pdf>). The abstract queries include aggregate functions to aggregate data values from data sources. (*See* Performing a simple aggregation on Query Builder columns, available at https://www2.microstrategy.com/producthelp/Current/ReportDesigner/WebHelp/Lang_1033/Content/performing_a_simple_aggregation_on_query_builder_columns.htm).

39. MicroStrategy Platform also includes MicroStrategy Intelligence Server which is the architectural foundation of the MicroStrategy platform and provides role-based access control for use with projects registered with Intelligence Server.

40. MicroStrategy Intelligence Server provides security roles for administrators, making it easy to assign privileges to users and delegate project administration duties. (*See, e.g.*, Privileges for Predefined Security Roles, available at https://www2.microstrategy.com/producthelp/current/SystemAdmin/WebHelp/Lang_1033/Content/Privileges_for_predefined_security_roles.htm; MicroStrategy Analytics and Mobility System Administration Guide, at 14, available at <https://www2.microstrategy.com/producthelp/manuals/en/Admin.pdf>).

COUNT I

(Infringement of U.S. Patent No. 8,341,172)

41. Daedalus incorporates by reference the allegations set forth in Paragraphs 1-43 of this Complaint as though fully set forth herein.

42. On information and belief, MicroStrategy has directly infringed and continues to directly infringe one or more claims of the '172 Patent, including at least claim 1 of the '172 Patent, in the state of Virginia, in this judicial district, and elsewhere in the United States by, among other things, making, using, selling, offering for sale, and/or importing into the United States products that embody one or more of the inventions claimed in the '172 Patent, including but not limited to the above-identified MicroStrategy Advanced Reporting Tools, including the exemplary MicroStrategy Platform, and all reasonably similar products ("the '172 Accused Products"), in violation of 35 U.S.C. § 271(a).

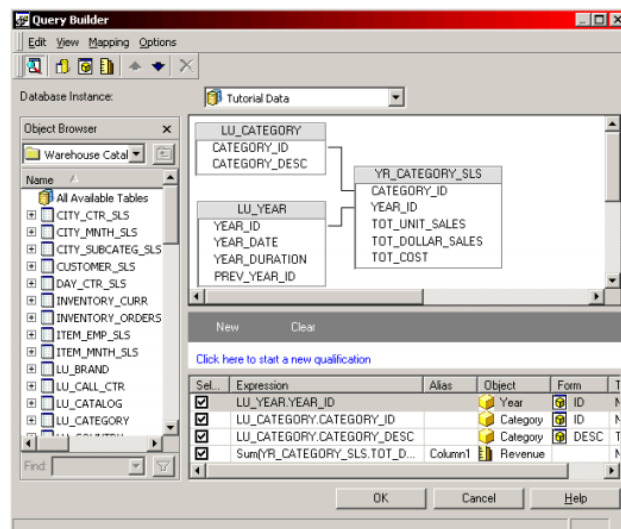
43. As an example, the '172 Accused Products comprise a "system for generating aggregate data values from data stored in a data source." (*See* MicroStrategy Cloud Platform Help, available at <https://www2.microstrategy.com/producthelp/current/Cloud/en-us/Content/home.htm>). Using Query Builder Editor, a graphical user interface, MicroStrategy Platform allows users to "aggregate functions to aggregate data values from data sources." (*See*,

e.g., Performing a simple aggregation on Query Builder columns, available at https://www2.microstrategy.com/producthelp/Current/ReportDesigner/WebHelp/Lang_1033/Content/performing_a_simple_aggregation_on_query_builder_columns.htm). Specifically, users can “perform a simple aggregation on a column expression in the Selections pane by right-clicking a column and pointing to Simple Aggregation, then selecting from the following aggregation functions.” (*See* Performing a simple aggregation on Query Builder columns). For example, users can “import data from a database, Hadoop or Google BigQuery.” (*How to Import Data from a database, Hadoop, or Google BigQuery by Building a SQL Query*, available at https://www2.microstrategy.com/producthelp/Current/Desktop/WebHelp/Lang_1033/Content/Importing_data_from_a_database_by_building_a_SQL_q.htm).

44. The MicroStrategy Platform includes a “processor.” For example, when subscribers host their data warehouse in MicroStrategy’s secure cloud, MicroStrategy maintains the software and hardware infrastructure, including a processor. (*See* MicroStrategy Cloud Platform Help).

45. MicroStrategy Platform provides “a database service available in a network environment.” (*See* How to Import Data from a database, Hadoop, or Google BigQuery by Building a SQL Query). For example, to import data from a relational data source, a communication must be established between MicroStrategy and a user’s data source. (*See* How to Create a Database Connection, available at https://www2.microstrategy.com/producthelp/Current/Desktop/WebHelp/Lang_1033/Content/creating_a_database_connection.htm). A database connection must be established to specify the connection information used to access the database. (*Id.*) This database service includes “a data source” and “an abstract data layer.” Specifically, users can import data directly from many “data sources” such as a database,

Hadoop, or Google BigQuery. (See How to Import Data from a Database, Hadoop, or Google BigQuery by Building a SQL Query). Furthermore, the Query Builder Editor is “an abstract data layer” which allows users to query the data sources without writing any SQL command. (See MicroStrategy Advanced Reporting Guide, at 773; See also:

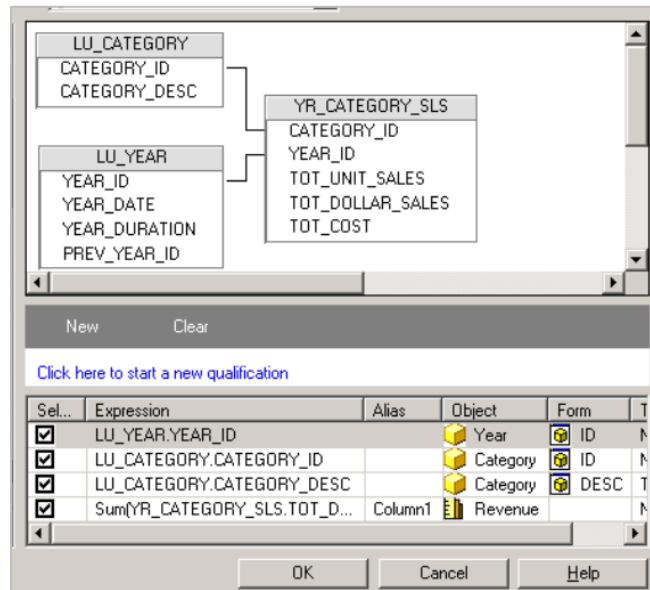


Id. at 774).

46. The Query Builder Editor provide a method of importing database tables which comprise of “a plurality of logical fields.” (See Advanced Reporting Guide, at 818). Using the Query Builder Editor, users can “compose an abstract query” to be run against imported database tables to “query the data source.” (See Advanced Reporting Guide, at 773-818). This provides an easy way to quickly access Open Database Connectivity data sources without having to write any SQL. (*Id.*).

47. The Query Builder Editor includes Conditions Pane and Selections Pane which are “access methods.” Conditions Pane allows users to define qualifications and insert security filters that will be included in the WHERE or HAVING clause of their SQL queries to filter the data that appears on the resulting report, which is “a method for accessing the data.” (See Advanced Reporting Guide, at 775). Similarly, Selections Pane also “specif[ies] a method for

accessing the data” by allowing users to select specific columns from the data source to be included in the SELECT clause of their SQL queries and become attributes and metrics on the resulting report. (*Id.*; see also:



Id. at 774). MicroStrategy Platform’s Advanced Reporting Tools allow users to aggregate data from a data source. In Query Builder Editor, “at least one logical field specifies an aggregate access method.” “You can perform a simple aggregation on a column expression in the Selections pane by right-clicking a column and pointing to Simple Aggregation, then selecting from the following aggregation functions.” (Performing a simple aggregation on Query Builder columns). For example, column expression can be modified to include a sum aggregation by right-clicking the specific column expression, point to simple aggregation, and select Sum. (*See* Advanced Reporting Guide, at 783).

48. Aggregate functions such as Sum are aggregate access methods “specify a set of input data and an expression for determining an aggregate value from the set of input data.” In MicroStrategy Platform’s Advanced Reporting Tools, the aggregate functions work on data values of a column of a database table which acts as the set of “input data.” The aggregate

function “Sum” is defined as a function that “[r]eturns the sum of all numbers in a list of values.”

(See Sum, available at https://www2.microstrategy.com/producthelp/current/FunctionsRef/Content/FuncRef/standard_functions_sum.htm). Aggregate values from all the data values of a column of the database table is calculated and a single result, or “an aggregate data value,” is returned. (See Performing a simple aggregation on Query Builder columns). The definition is thus “an expression for determining an aggregate value from the set of input data.”

MicroStrategy Platform’s Advanced Reporting Tools includes “a runtime component configured to process an abstract query that includes at least one logical field.” (See Query Builder Editor, available at https://www2.microstrategy.com/producthelp/Current/ReportDesigner/WebHelp/Lang_1033/Content/query_builder_editor.htm).

49. Using MicroStrategy Platform’s Advanced Reporting Tools, users click the Run Report icon, “a runtime component,” to run the Query Builder report. (*Id.*) The Run Report icon is “configured to process an abstract query that includes the at least one logical field.” For instance, Query Builder reports require users to first define a valid query to run against their data source. Users must also map objects to the data returned by their query. Once such tasks are completed, users can begin reporting and analyzing the data returned from their query with the Reporter Editor. (*Id.*) The queries generated by Query Builder report include database tables which are “logical fields.”

50. Using MicroStrategy Platform’s Advanced Reporting Tools, when users click the Run Report icon to run an aggregate function such as Sum, the Query Builder “retrieves a definition for the aggregate access method.” The descriptions of the expressions are “definitions of the aggregate access method.” For example, when a user updates the expression Sum (DAY_CTR_SLS.TOT_COST) to Sum (DAY_CTR_SLS.TOT_COST*1.06), MicroStrategy

updates the mapping between the query and the data sources automatically to be associated with the new expression. (*See* MicroStrategy Advanced Reporting Guide, at 794-797).


51. Using MicroStrategy Platform's Advanced Reporting Tools, when users click the Run Report icon to run an aggregate function such as Sum, the Query Builder further "determin[es] aggregate data values according to the definition" of the aggregate access method. For example, the aggregate function "Sum" is defined as a function that "[r]eturns the sum of all numbers in a list of values." (*See* Sum). Aggregate values from all the data values of a column of the database table is calculated and a single result, or "an aggregate data value," is returned. (*See* Performing a simple aggregation on Query Builder columns). The aggregate function Sum is used to calculate the summation of the data values represented by a column of a database table.

52. In MicroStrategy Platform's Advanced Reporting Tools, the Query Builder "merg[es] the aggregate data values with query results obtained for logical fields, other than the at least one logical field, included in the abstract query." For example, results of summation in the column "Revenue," which are "aggregate data values," are merged with the values in the columns "Year," "Category," and "Metrics," which are "query results obtained for logical fields, other than the at least one logical field, included in the abstract query." (*See* MicroStrategy Advanced Reporting Guide, at 783; *see also*:

Select	Expression	Alias	Object	Form	Type
<input checked="" type="checkbox"/>	LU_YEAR.YEAR_ID		Year	ID	Number
<input checked="" type="checkbox"/>	LU_CATEGORY.CATEGORY_ID		Category	ID	Number
<input checked="" type="checkbox"/>	LU_CATEGORY.CATEGORY_DESC		Category	DESC	Text
<input checked="" type="checkbox"/>	YR_CATEGORY_SLS.TOT_DOLLAR_S...		Revenue		Number

This report is created within the MicroStrategy Tutorial project, therefore the column expressions have been automatically mapped to the existing metadata objects. In a project without attribute or metric definitions, you must manually enter attribute and metric names, forms, and data types to be associated with each column expression. For more information on mapping columns to different metadata objects, see the section [Map Columns to Metadata Objects, page 795](#).

- Right-click the **YR_CATEGORY_SLS.TOT_DOLLAR_SALES** column expression, point to **Simple Aggregation**, and select **Sum**. This action modifies the column expression to include a sum aggregation.
- Click **OK**.

 You can define the Query Builder report in the same way you define a standard report, adding features such as formatting, sorting, view filters, thresholds, exporting, and so on.

- Save the Query Builder report as **Query Builder Demo**.

 You must save the report first before you can run it.

This report is created within the MicroStrategy Tutorial project, therefore the column expressions have been automatically mapped to the existing metadata objects. In a project without attribute or metric definitions, you must manually enter attribute and metric names, forms, and data types to be associated with each column expression. For more information on mapping columns to different metadata objects, see the section [Map columns to metadata objects, page 593](#).

- Right-click the **YR_CATEGORY_SLS.TOT_DOLLAR_SALES** column expression, point to **Simple Aggregation**, and select **Sum**. This action modifies the column expression to include a sum aggregation.
- Click **OK** to exit the Query Builder Editor. The Report Editor is displayed.

You can define the Query Builder report in the same way you define a standard report, adding features such as formatting, sorting, view filters, thresholds, exporting, and so on.

- Save the Query Builder report as **Query Builder Demo**.

 You must save the report first before you can run it.

- Run the report. The image below shows the report in grid view:

Year	Category	Metrics	Revenue
2006	Books		650,192
	Electronics		6,027,843
	Movies		1,012,594
	Music		956,610
2007	Books		868,207
	Electronics		8,020,662
	Movies		1,345,502
	Music		1,283,235
2008	Books		1,121,696
	Electronics		10,342,798
	Movies		1,740,847
	Music		1,653,523

Notice that revenue data is displayed with a general format by default. You can format the Revenue metric's values to apply a currency format to the revenue data.

Id at 783-784.). In the example provided by MicroStrategy Advanced Reporting Guide, the CATEGORY_ID from LU_CATEGORY table and CATEGORY_DESC from

LU_CATEOGRY table are “logical fields, other than the at least one logical field.” (*Id.* at 782-784).

53. In MicroStrategy Platform’s Advanced Reporting Tools, after processing the query generated by Query Builder Editor, the Query Builder report “returns the results to the requesting entity.” Specifically, once a user defines a valid query to run against his data source and maps objects to the data returned by his query, the user can begin reporting and analyzing the data returned from his query with the Reporter Editor. (*See* Building reports with Query Builder, available at https://www2.microstrategy.com/producthelp/Current/ReportDesigner/WebHelp/Lang_1033/Content/overview_of_query_builder_reports.htm; *see also*:

Year	Category	Metrics	Revenue
2006	Books		650,192
	Electronics		6,027,843
	Movies		1,012,594
	Music		956,610
2007	Books		868,207
	Electronics		8,020,662
	Movies		1,345,502
	Music		1,263,235
2008	Books		1,121,696
	Electronics		10,342,798
	Movies		1,740,847
	Music		1,853,523

MicroStrategy Advanced Reporting Guide, at 784).

54. By making, using, offering for sale, and/or selling products in the United States and/or importing products into the United States, including but not limited to the ’172 Accused Products, MicroStrategy has injured Daedalus and is liable to Daedalus for directly infringing one or more claims of the ’172 Patent, including without limitation claim 1 pursuant to 35 U.S.C. § 271(a).

55. On information and belief, MicroStrategy will continue to infringe the ’172 Patent unless enjoined by this Court.

56. As a result of MicroStrategy's infringement of the '172 Patent, Daedalus has suffered monetary damages, and seeks recovery, in an amount to be proven at trial, adequate to compensate for MicroStrategy's infringement, but in no event less than a reasonable royalty with interest and costs. MicroStrategy's infringement of Daedalus' rights under the '172 Patent will continue to damage Daedalus, causing irreparable harm for which there is no adequate remedy at law, unless enjoined by this Court.

COUNT II

(Infringement of U.S. Patent No. 9,032,076)

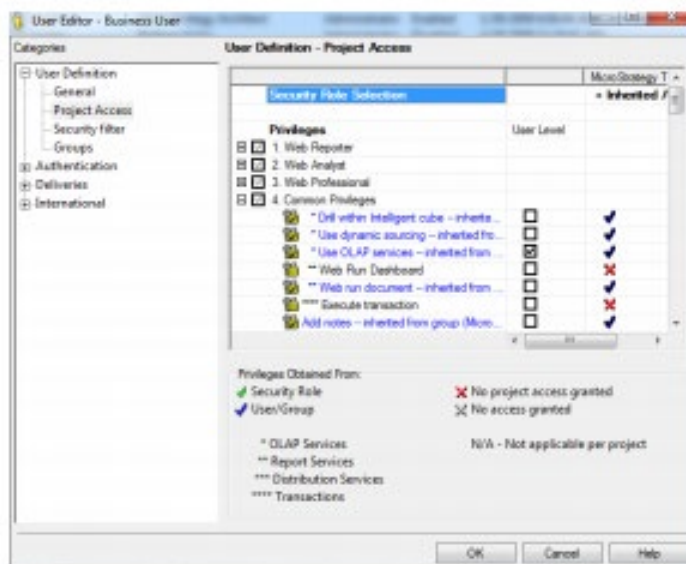
57. Daedalus incorporates by reference the allegations set forth in Paragraphs 1- 59 of this Complaint as though fully set forth herein.

58. On information and belief, MicroStrategy has directly infringed and continues to directly infringe one or more claims of the '076 Patent, including at least Claim 6 of the '076 Patent, in the state of Virginia, in this judicial district, and elsewhere in the United States by, among other things, making, using, selling, offering for sale, and/or importing into the United States products that embody one or more of the inventions claimed in the '076 Patent, including but not limited to the above-identified MicroStrategy Platform which includes MicroStrategy Intelligence Server, and all reasonably similar products ("the '076 Accused Products"), in violation of 35 U.S.C. § 271(a).

59. As an example, the '076 Accused Products, including MicroStrategy Platform with Intelligence Server, implement "a role-based access control method" to prevent unauthorized access to project resources by precisely granting actions users can perform. "Beginning with version 9.0, MicroStrategy product suite comes with a number of predefined

security roles for administrators. These roles make it easy to delegate administrative tasks.”

(Defining sets of privileges: Security roles; *see also*:

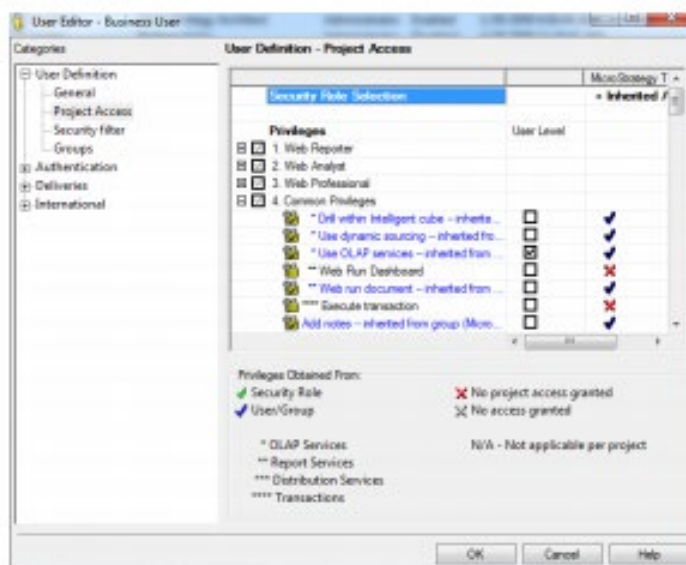


Platform Security, at 21, available at <https://custom.cvent.com/83AD54AE0B7F4CBC94A9A945D0A278DA/files/e98e6b4bbca14beda0c92b3a3572c1e3.pdf>).

60. MicroStrategy Intelligence Server’s role-based access control method includes “defining roles to be sets of permissions on individual resources, thus forming role instances.” For example, users or groups are assigned roles such as Web Reporter, and roles are assigned privileges to access protected resources and perform actions. That is, users or groups gain access and permission to perform actions when the user is assigned a role. (*See Platform Security*, at 21). In MicroStrategy Platform, when privileges are added to a role such as Web Reporter, thereby binding resources with the Developer role, a “role instance” is formed. “A security role is a collection of privileges in a project.” (Defining sets of privileges: Security roles).

61. MicroStrategy Platform includes Intelligence Server which “assign[s] at least one set of role instances to at least one group.” For example, “A security role is a collection of

project-level privileges that are assigned to users and groups.” (How to Assign Access Privileges to a Group, available at https://www2.microstrategy.com/producthelp/current/WebAdmin/WebHelp/Lang_1033/Content/Assign_privileges_to_group.htm). “To assign a security role to the group open the drop-down list and select the checkbox next to the role or roles you want to assign.” (Group Editor: Project Access Tab, available at https://www2.microstrategy.com/producthelp/current/WebAdmin/WebHelp/Lang_1033/Content/Group_Editor_Project_Access.htm; *see also*:



Platform Security, at 21).

62. MicroStrategy Platform with Intelligence Server further “assign[s] at least one group of a set of role instances to at least one super role.” For instance, Web Analyst and Web Professional are “super roles.” MicroStrategy Intelligence Server assigns security roles which are “role instances” to Web Analyst and Web Professional roles. (*See, e.g.,* Group Editor: Project Access Tab; Privileges for Predefined Security Roles: Security roles). Users or groups are assigned Web Analyst and Web Professional roles, which inherit the privileges and permissions of Web Reporter. (*See* Web Analyst Privileges, available at

https://www2.microstrategy.com/producthelp/current/SystemAdmin/WebHelp/Lang_1033/Content/Web_Analyst_privileges.htm; Web Professional Privileges, available at https://www2.microstrategy.com/producthelp/current/SystemAdmin/WebHelp/Lang_1033/Content/Web_Professional_privileges.htm).

63. In MicroStrategy Platform with Intelligence Server, a super role such Web Professional, is “nest[ed...] according to ... properties including a name, a parent role, the set of role instances.” For instance, the Web Analyst role must inherit, or are nested according to, privileges and security roles of Web reporter. (See Web Analyst Privileges). Additionally, Web Professional roles must inherit the privileges and security roles of Web Analyst, a “parent role,” and Web Reporter. (See Web Professional Privileges; *see also* Group Editor: Project Access Tab; *see also*:



Platform Security, at 24).

64. Additionally, a super role such as Web Professional, is “nest[ed...] according to ... an externalisation state.” An administrator uses Intelligence Server Administrator Page, which requires an external authentication to be performed during login, to manage users and groups. (See Intelligence Server Administrator Page, available at https://www2.microstrategy.com/producthelp/current/SystemAdmin/WebHelp/Lang_1033/Content/Web_Analyst_privileges.htm).

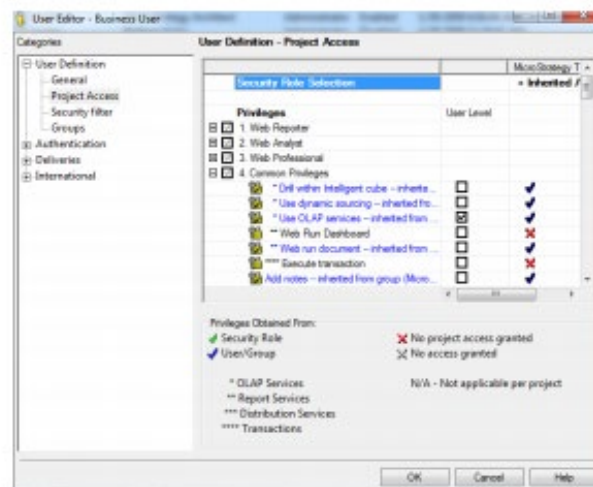
com/producthelp/current/WebAdmin/WebHelp/Lang_1033/Content/IServer_Admin.htm). A user must enter his or her administrative login and password to log in to the Intelligence Server Administrator Page. (*Id.*; *see also*:

To View a User's Privileges

1. In Developer, log in to a project source. You must log in as a user with the Create And Edit Users And Groups privilege.
2. Expand **Administration**, then **User Manager**, and then the group containing the user.
3. Right-click the user and select **Grant access to projects**. The User Editor opens to the Project Access dialog box. The privileges that the user has for each project are listed, as well as the source of those privileges (inherent to user, inherited from a group, or inherited from a security role).

MicroStrategy Analytics and Mobility System Administration Guide, at 87).

65. MicroStrategy Platform with Intelligence Server provides for each super role to be “modified by adding or removing the role instances from at least one group.” For instance, “[t]o assign a security role to the group open the drop-down list and select the checkbox next to the role or roles you want to assign.” (Group Editor: Project Access Tab; *see also*:



Platform Security, at 21). Additionally, security roles can be deleted from the Intelligence Server Administrator page. (*See* Deleting a Security Role, available at https://www2.microstrategy.com/producthelp/current/WebAdmin/WebHelp/Lang_1033/Content/Delete_security_role.htm).

66. By making, using, offering for sale, and/or selling products in the United States and/or importing products into the United States, including but not limited to the '076 Accused Products, MicroStrategy has injured Daedalus and is liable to Daedalus for directly infringing one or more claims of the '076 Patent, including without limitation claim 6 pursuant to 35 U.S.C. § 271(a).

67. On information and belief, MicroStrategy will continue to infringe the '076 Patent unless enjoined by this Court.

68. As a result of MicroStrategy's infringement of the '076 Patent, Daedalus has suffered monetary damages, and seeks recovery, in an amount to be proven at trial, adequate to compensate for MicroStrategy's infringement, but in no event less than a reasonable royalty with interest and costs. MicroStrategy's infringement of Daedalus' rights under the '076 Patent will continue to damage Daedalus, causing irreparable harm for which there is no adequate remedy at law, unless enjoined by this Court.

PRAYER FOR RELIEF

WHEREFORE, Plaintiff prays for judgment and *seeks* relief against MicroStrategy as follows:

- a. For judgment that MicroStrategy has infringed and continues to infringe the claims of the '172 and '076 Patents;

- b. For a permanent injunction against MicroStrategy and its respective officers, directors, agents, servants, affiliates, employees, divisions, branches, subsidiaries, parents, and all other acting in active concert therewith from infringement of the '172 and '076 Patents;
- c. For an Award of the damages sustained by Plaintiff as the result of MicroStrategy's acts of infringement;
- d. For a mandatory future royalty payable on each and every future sale by MicroStrategy of a product that is found to infringe one or more of the Daedalus Patents and on all future products which are not colorably different from products found to infringe;
- e. For a judgment and order requiring MicroStrategy to pay Plaintiff's damages, costs, expenses, and pre- and post-judgment interest for its infringement of the '172 and '076 Patents as provided under 35 U.S.C. § 284; and
- f. For such other and further relief in law and in equity as the Court may deem just and proper.

DEMAND FOR JURY TRIAL

Pursuant to Rule 38(b) of the Federal Rules of Civil Procedure, Plaintiff demands a trial by jury in this action of all issues triable by a jury.

Dated: November 4, 2020

Respectfully Submitted,

/s/ Walter D. Kelley, Jr. _____

Walter D. Kelley, Jr. (VSB No. 21622)
HAUSFELD, LLP
888 16th Street, N.W., Suite 300
Washington, DC 20006
Tel: (202) 540-7157
Fax: (202) 540-7201
E-mail: wkelley@hausfeld.com

Denise M. De Mory (*Pro Hac Pending*)
Cal. Bar No. 168076
ddemory@bdiplaw.com
Jennifer L. Gilbert (*Pro Hac Pending*)
Cal. Bar. No. 255820
jgilbert@bdiplaw.com
Robin Curtis (*Pro Hac Pending*)
Cal. Bar No. 271702
rcurtis@bdiplaw.com
Gail Jefferson (*Pro Hac Pending*)
Cal. Bar No. 325874
gjefferson@bdiplaw.com
BUNSOW DE MORY LLP
701 El Camino Real
Redwood City, CA 94063
Telephone: (650) 351-7248
Facsimile: (415) 426-4744

Attorneys for Plaintiff
Daedalus Blue, LLC